

# ЛАБОРАТОРНА РОБОТА № 3

## ДОСЛІДЖЕННЯ МЕТОДІВ РЕГРЕСІЇ ТА НЕКОНТРОЛЬОВАНОГО НАВЧАННЯ

**Мета роботи:** використовуючи спеціалізовані бібліотеки і мову програмування Python дослідити методи регресії та неконтрольованої класифікації даних у машинному навчанні.

### Завдання 2.1 Створення регресора однієї змінної.

```
1 import pickle
2 import numpy as np
3 from sklearn import linear_model
4 import sklearn.metrics as sm
5 import matplotlib.pyplot as plt
6
7 input_file = 'data_singlevar_regr.txt'
8 # Завантаження даних
9 data = np.loadtxt(input_file, delimiter=',')
10 X, y = data[:, :-1], data[:, -1]
11 # Розбивка даних на навчальний та тестовий набори
12 num_training = int(0.8 * len(X))
13 num_test = len(X) - num_training
14 # Тренувальні дані
15 X_train, y_train = X[:num_training], y[:num_training]
16 # Тестові дані
17 X_test, y_test = X[num_training:], y[num_training:]
18 # Створення об'єкта лінійного регресора
19 regressor = linear_model.LinearRegression()
20 # Тренування моделі
21 regressor.fit(X_train, y_train)
22 # Прогнозування результату
23 y_test_pred = regressor.predict(X_test)
24 # Побудова графіка
25 plt.scatter(X_test, y_test, color='green')
26 plt.plot(X_test, y_test_pred, color='black', linewidth=4)
27 plt.xticks(())
28 plt.yticks(())
29 plt.show()
30 # Обрахування метрик
31 print("Linear Regression performance:")
```

					ДУ «Житомирська політехніка».20.121.12.			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Надворний М.Ю.						
Перевір.		Філіпов В.О.						
Керівник								
Н. контр.								
Зав. каф.								
						Літ.	Арк.	Аркушів
							1	17
						ФІКТ Гр. ІПЗк-20-1		

```

# Обрахування метрик
print("Linear regressor performance:")
print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2))
print("Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, y_test_pred), 2))
print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
# Файл для збереження моделі
output_model_file = 'model.pkl'
# Збереження моделі
with open(output_model_file, 'wb') as f:
    pickle.dump(regressor, f)
# Завантаження моделі
with open(output_model_file, 'rb') as f:
    regressor_model = pickle.load(f)
y_test_pred_new = regressor_model.predict(X_test)
print("\nNew mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred_new), 2))

```

Рис 1. Код програми файлу LR\_3\_task\_1m

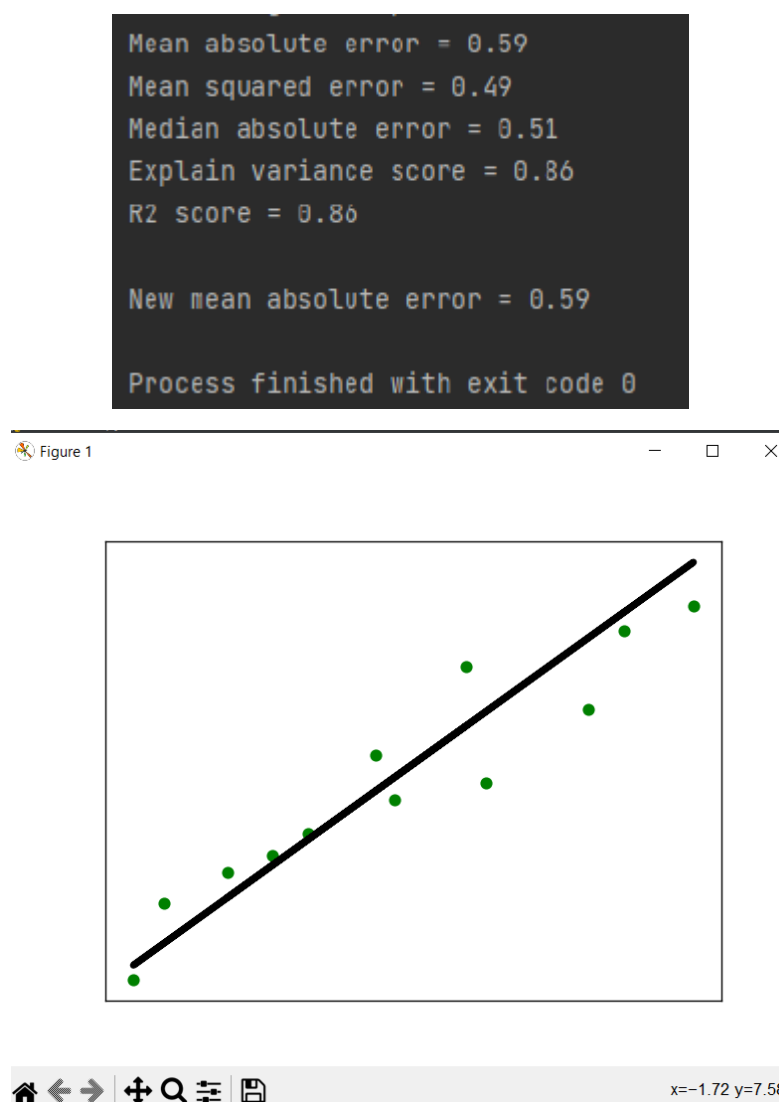


Рис 2. Результат виконання коду файлу LR\_3\_task\_1m

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				2
Змн.	Арк.	№ докум.	Підпис	Дата		

**Висновок:** модель для вихідних даних побудована валідно. MAE, MSE – середня якість. Показник R2 – добре.

**Завдання 2.2.** Передбачення за допомогою регресії однієї змінної.

Номер – 12

Варіант – 2

№ за списком	11	12	13	14	15	16	17	18	19	20
№ варіанту	1	2	3	4	5	1	2	3	4	5

```

1 import pickle
2 import numpy as np
3 from sklearn import linear_model
4 import sklearn.metrics as sm
5 import matplotlib.pyplot as plt
6
7 # Вхідний файл, який містить дані
8 input_file = 'data_regr_2.txt'
9 # Завантаження даних
10 data = np.loadtxt(input_file, delimiter=',')
11 X, y = data[:, :-1], data[:, -1]
12 # Розбивка даних на навчальний та тестовий набори
13 num_training = int(0.8 * len(X))
14 num_test = len(X) - num_training
15 # Тренувальні дані
16 X_train, y_train = X[:num_training], y[:num_training]
17 # Тестові дані
18 X_test, y_test = X[num_training:], y[num_training:]
19 # Створення об'єкта лінійного регресора
20 regressor = linear_model.LinearRegression()
21 # Тренування моделі
22 regressor.fit(X_train, y_train)
23 # Прогнозування результату
24 y_test_pred = regressor.predict(X_test)
25 # Побудова графіка
26 plt.scatter(X_test, y_test, color='green')
27 plt.plot(X_test, y_test_pred, color='black', linewidth=4)
28 plt.xticks(())
29 plt.yticks(())
30 plt.show()

```

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				3
Змн.	Арк.	№ докум.	Підпис	Дата		

```

print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
# Файл для збереження моделі
output_model_file = 'model.pkl'
# Збереження моделі
with open(output_model_file, 'wb') as f:
    pickle.dump(regressor, f)
# Завантаження моделі
with open(output_model_file, 'rb') as f:
    regressor_model = pickle.load(f)

y_test_pred_new = regressor_model.predict(X_test)
print("\nNew mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred_new), 2))

```

Рис 3. Код програми файлу LR\_3\_task\_2m

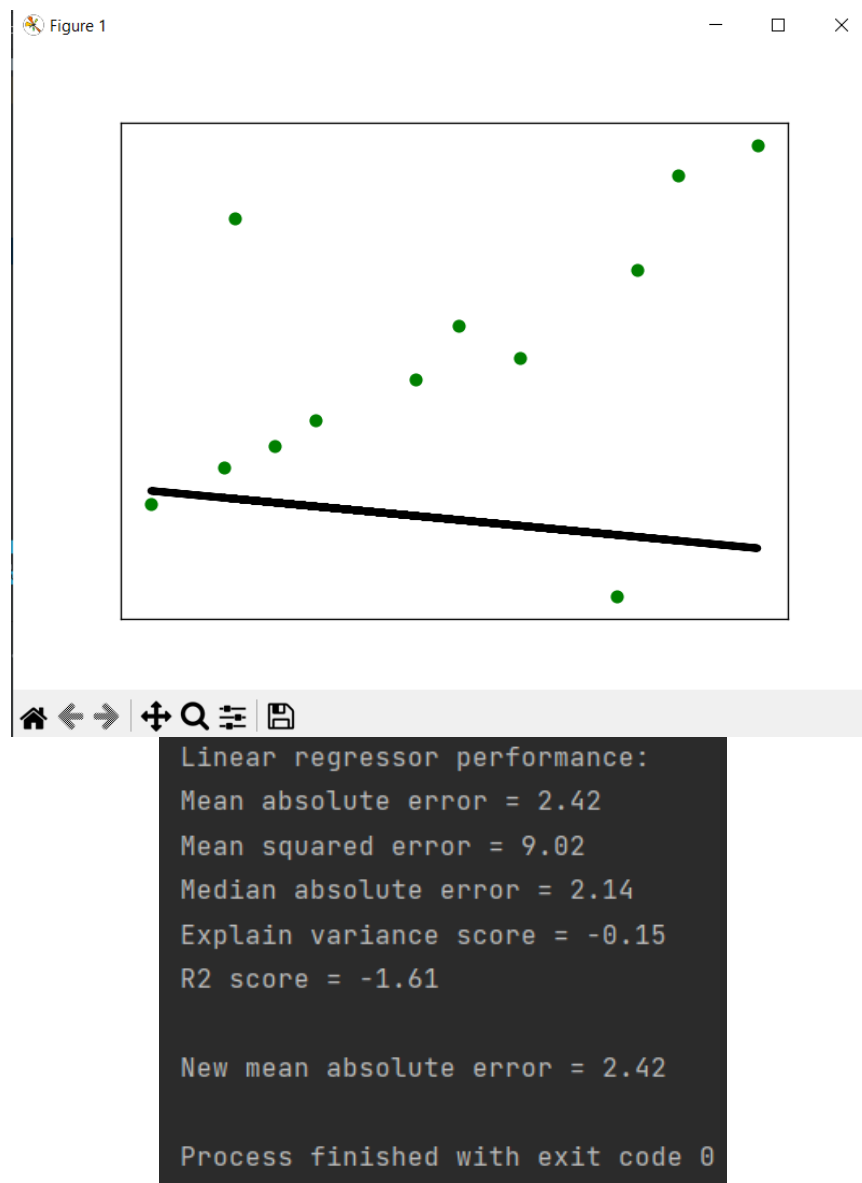


Рис 4. Результат виконання коду файлу LR\_3\_task\_2m

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				4
Змн.	Арк.	№ докум.	Підпис	Дата		

### Завдання 2.3. Створення багатовимірної регресора.

```

1 import numpy as np
2 from sklearn import linear_model
3 import sklearn.metrics as sm
4 from sklearn.preprocessing import PolynomialFeatures
5 # Вхідний файл, який містить дані
6 input_file = 'data_multivar_regr.txt'
7 # Завантаження даних
8 data = np.loadtxt(input_file, delimiter=',')
9 X, y = data[:, :-1], data[:, -1]
10 # Розбивка даних на навчальний та тестовий набори
11 num_training = int(0.8 * len(X))
12 num_test = len(X) - num_training
13 # Тренувальні дані
14 X_train, y_train = X[:num_training], y[:num_training]
15 # Тестові дані
16 X_test, y_test = X[num_training:], y[num_training:]
17 # Створення об'єкта лінійного регресора
18 linear_regressor = linear_model.LinearRegression()
19 # Тренування моделі
20 linear_regressor.fit(X_train, y_train)
21 # Прогнозування результату
22 y_test_pred = linear_regressor.predict(X_test)
23 # Обрахування метрик
24 print("Linear Regressor performance:")
25 print("Mean absolute error =", round(sm.mean_absolute_error(y_test, y_test_pred), 2))
26 print("Mean squared error =", round(sm.mean_squared_error(y_test, y_test_pred), 2))
27 print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
28 print("Explained variance score =", round(sm.explained_variance_score(y_test, y_test_pred), 2))
29 print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
30 # Поліноміальна регресія
31 polynomial = PolynomialFeatures(degree=10)
32 X_train_transformed = polynomial.fit_transform(X_train)
33 datapoint = [[7.75, 6.35, 5.56]]
34 poly_datapoint = polynomial.fit_transform(datapoint)
35
36 poly_linear_model = linear_model.LinearRegression()
37 poly_linear_model.fit(X_train_transformed, y_train)
38 print("\nLinear regression:\n", linear_regressor.predict(datapoint))
39 print("\nPolynomial regression:\n", poly_linear_model.predict(poly_datapoint))

```

Рис 5. Код програми файлу LR\_3\_task\_3m

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

```

Linear Regressor performance:
Mean absolute error = 3.58
Mean squared error = 20.31
Median absolute error = 2.99
Explained variance score = 0.86
R2 score = 0.86

Linear regression:
[36.05286276]

Polynomial regression:
[41.46007151]

Process finished with exit code 0

```

Рис 6. Результат виконання коду файлу LR\_3\_task\_3

**Висновок:** Порівнюючи з лінійним регресором, поліноміальний регресор більш кращий, тобто дозволяє показувати кращі результати. На це вказує значення 41.45.

## Завдання 2.4. Регресія багатьох змінних.

```

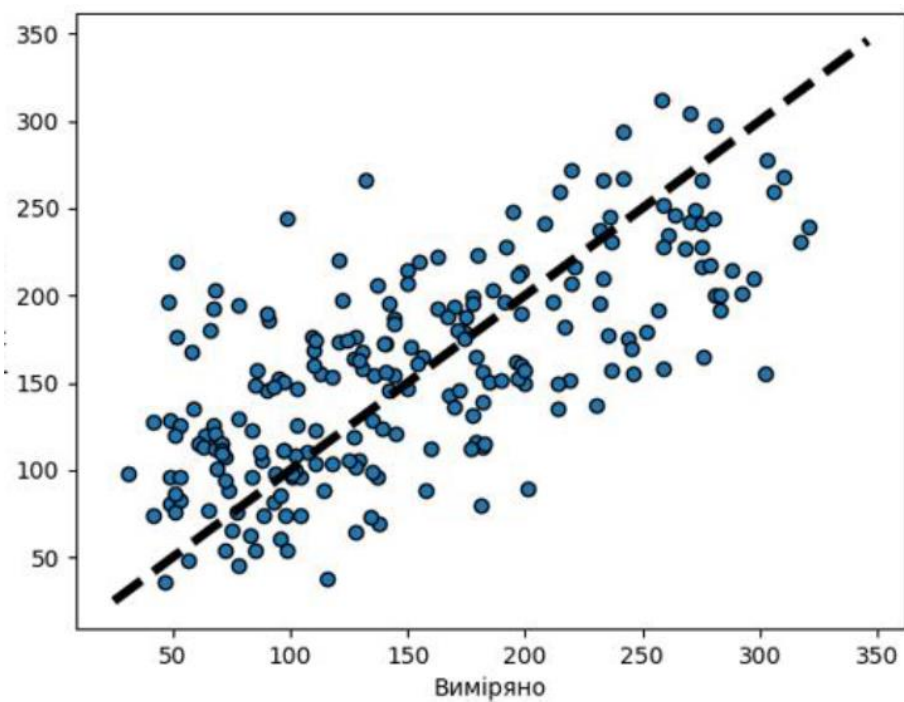
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import train_test_split

diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size=0.5, random_state=0)
regr = linear_model.LinearRegression()
regr.fit(Xtrain, ytrain)
ypred = regr.predict(Xtest)
# Обрахування метрик
print("regr.coef =", np.round(regr.coef_, 2))
print("regr.intercept =", round(regr.intercept_, 2))
print("R2 score =", round(r2_score(ytest, ypred), 2))
print("Mean absolute error =", round(mean_absolute_error(ytest, ypred), 2))
print("Mean squared error =", round(mean_squared_error(ytest, ypred), 2))
fig, ax = plt.subplots()
ax.scatter(ytest, ypred, edgecolors=(0, 0, 0))
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Виміряно')
ax.set_ylabel('Предбачено')
plt.show()

```

Рис 7. Код програми файлу LR\_3\_task\_4

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				6
Змн.	Арк.	№ докум.	Підпис	Дата		



```

regr.coef = [ -20.4  -265.89  564.65  325.56 -692.16  395.56  23.5  116.36  843.95
              12.72]
regr.intercept = 154.36
R2 score = 0.44
Mean absolute error = 44.8
Mean squared error = 3075.33

Process finished with exit code 0

```

Рис 8. Результат виконання коду файлу LR\_3\_task\_5

### Завдання 2.5. Самостійна побудова регресії.

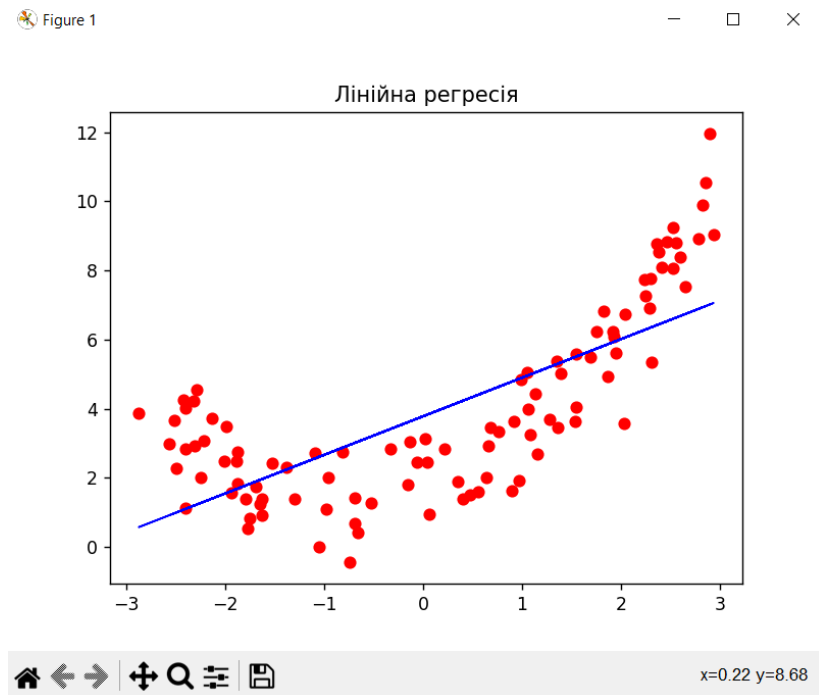
№ за списком	11	12	13	14	15	16	17	18	19	20
№ варіанту	1	2	3	4	5	6	7	8	9	10

```

1 import numpy as np
2 from matplotlib import pyplot as plt
3 from sklearn import linear_model
4 import sklearn.metrics as sm
5 from sklearn.preprocessing import PolynomialFeatures
6 # Генерація даних
7 m = 100
8 X = 6 * np.random.rand(m, 1) - 3
9 y = 0.6 * X ** 2 + X + 2 + np.random.randn(m, 1)
10 X = X.reshape(-1, 1)
11 y = y.reshape(-1, 1)
12 # Лінійна регресія
13 linear_regressor = linear_model.LinearRegression()
14 linear_regressor.fit(X, y)
15 # Поліноміальна регресія
16 polynomial = PolynomialFeatures(degree=2, include_bias=False)
17 X_poly = polynomial.fit_transform(X)
18 polynomial.fit(X_poly, y)
19 poly_linear_model = linear_model.LinearRegression()
20 poly_linear_model.fit(X_poly, y)
21 y_pred = poly_linear_model.predict(X_poly)
22 print("\nr2: ", sm.r2_score(y, y_pred))
23 # Лінійна регресія
24 plt.scatter(X, y, color='red')
25 plt.plot(X, linear_regressor.predict(X), color='blue', linewidth=1)
26 plt.title("Лінійна регресія")
27 plt.show()
28 # Поліноміальна регресія
29 plt.scatter(X, y, color='red')
30 plt.plot(X, y_pred, "+", color='blue', linewidth=2)
31 plt.title("Поліноміальна регресія")

```

Рис 9. Код програми файлу LR\_3\_task\_5



		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				8
Змн.	Арк.	№ докум.	Підпис	Дата		



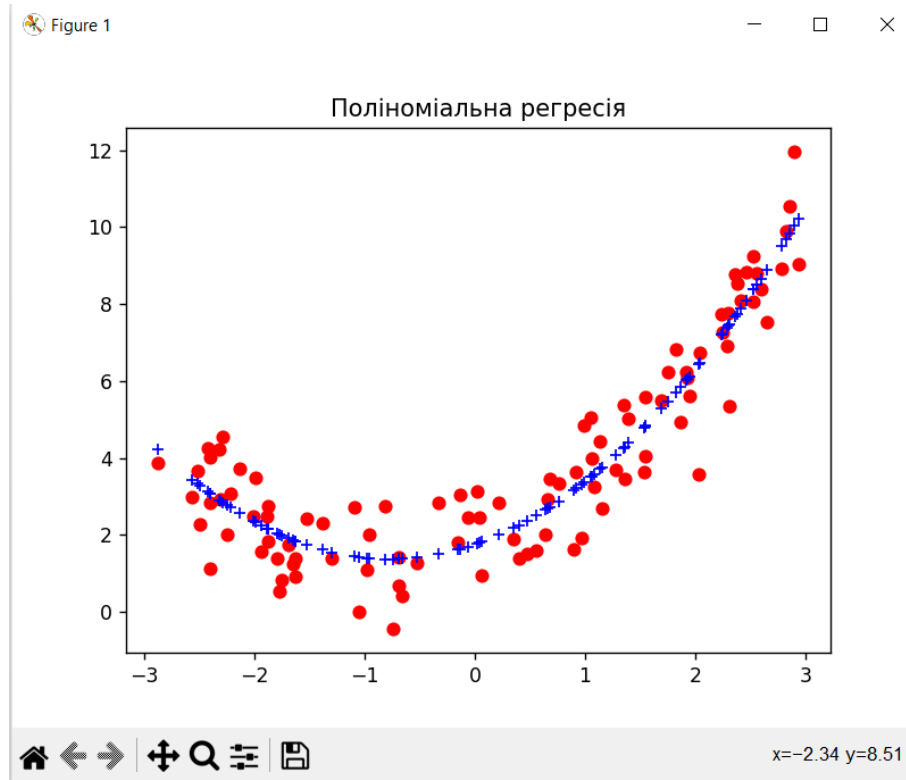


Рис 10. Результат виконання коду файлу LR\_3\_task\_5

### Завдання 2.6. Побудова кривих навчання.

№ за списком	11	12	13	14	15	16	17	18	19	20
№ варіанту	1	2	3	4	5	6	7	8	9	10

```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 from sklearn import linear_model
4 from sklearn.metrics import mean_squared_error
5 from sklearn.model_selection import train_test_split
6 from sklearn.preprocessing import PolynomialFeatures
7 # Генерація даних
8 m = 100
9 X = 6 * np.random.rand(m, 1) - 3
10 y = 0.6 * X ** 2 + X + 2 + np.random.randn(m, 1)
11
12 def plot_learning_curves(model, X, y):
13     X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2)
14     train_errors, val_errors = [], []
15     for m in range(1, len(X_train)):
16         model.fit(X_train[:m], y_train[:m])
17         y_train_predict = model.predict(X_train[:m])
18         y_val_predict = model.predict(X_val)
19         train_errors.append(mean_squared_error(y_train_predict, y_train[:m]))
20         val_errors.append(mean_squared_error(y_val_predict, y_val))
21     plt.plot(np.sqrt(train_errors), "r-+", linewidth=2, label='train')
22     plt.plot(np.sqrt(val_errors), "b-", linewidth=3, label='val')
23     plt.legend()
24     plt.show()
25
26 lin_reg = linear_model.LinearRegression()
27 plot_learning_curves(lin_reg, X, y)
28
29 from sklearn.pipeline import Pipeline
30 polynomial_regression = Pipeline([
31     ("poly_features",
32      PolynomialFeatures(degree=10, include_bias=False)),
33     ("lin_reg", linear_model.LinearRegression())
34 ])
35
36 plot_learning_curves(polynomial_regression, X, y)

```

Рис 11. Код програми файлу LR\_3\_task\_6

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

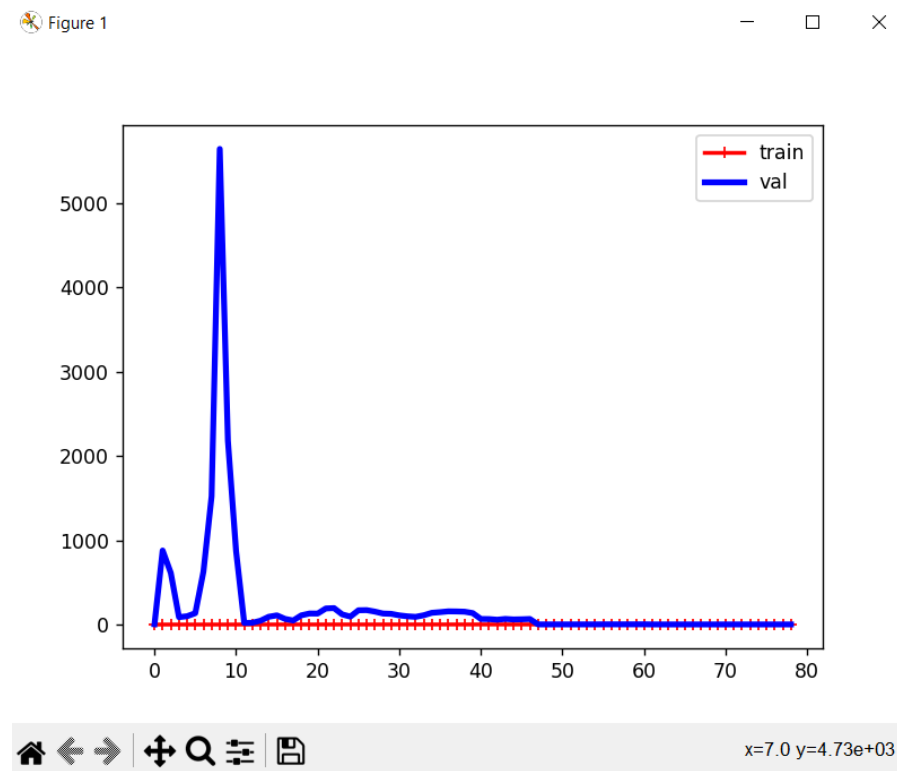
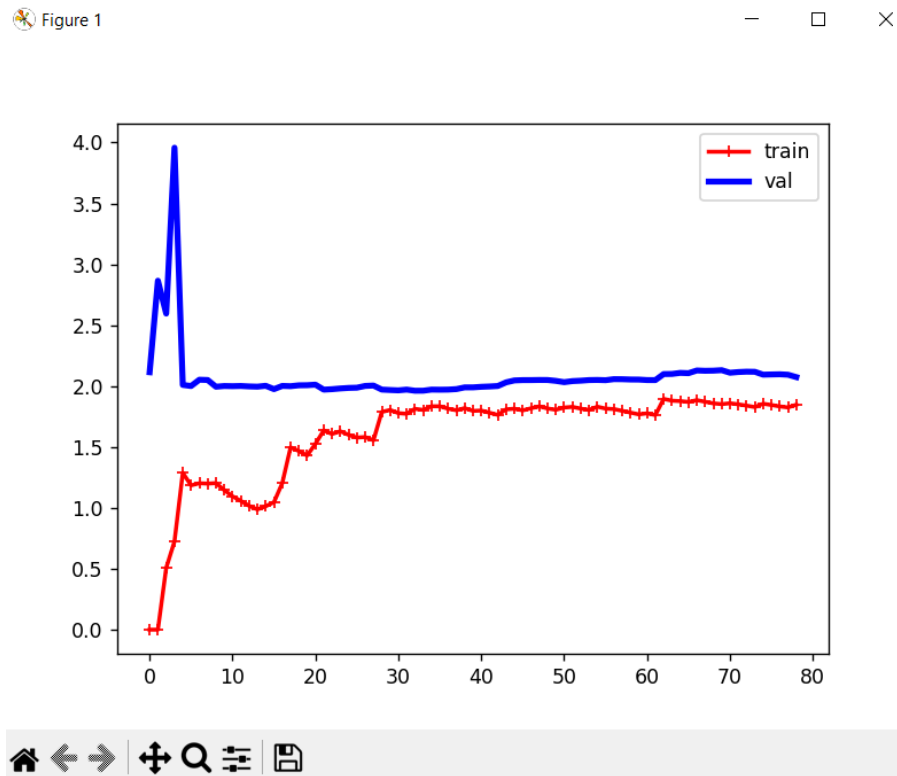


Рис 12. Результат виконання коду файлу LR\_3\_task\_6

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

**Висновок:** для з'ясування ступеня складності необхідної моделі ми можемо використати криві навчання. Для досягнення успіху необхідно досягти компромісу між зміщенням та дисперсією. В нашому випадку найкращий результат показала модель 2 ступеня.

**Завдання 2.7.** Кластеризація даних за допомогою методу k-середніх.

Рис 13. Код програми файлу LR\_3\_task\_7

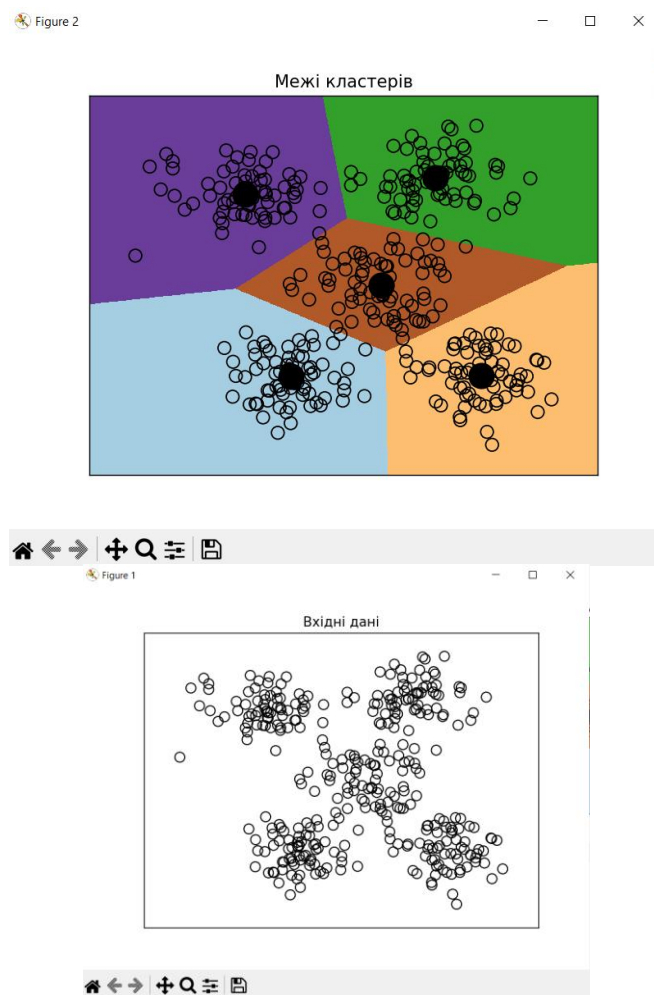


Рис 14. Результат виконання коду файлу LR\_3\_task\_7

**Висновок:** метод k-середніх валідно працює, але за умови, відомої кількості кластерів.

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

## Завдання 2.8. Кластеризація К-середніх для набору даних Iris.

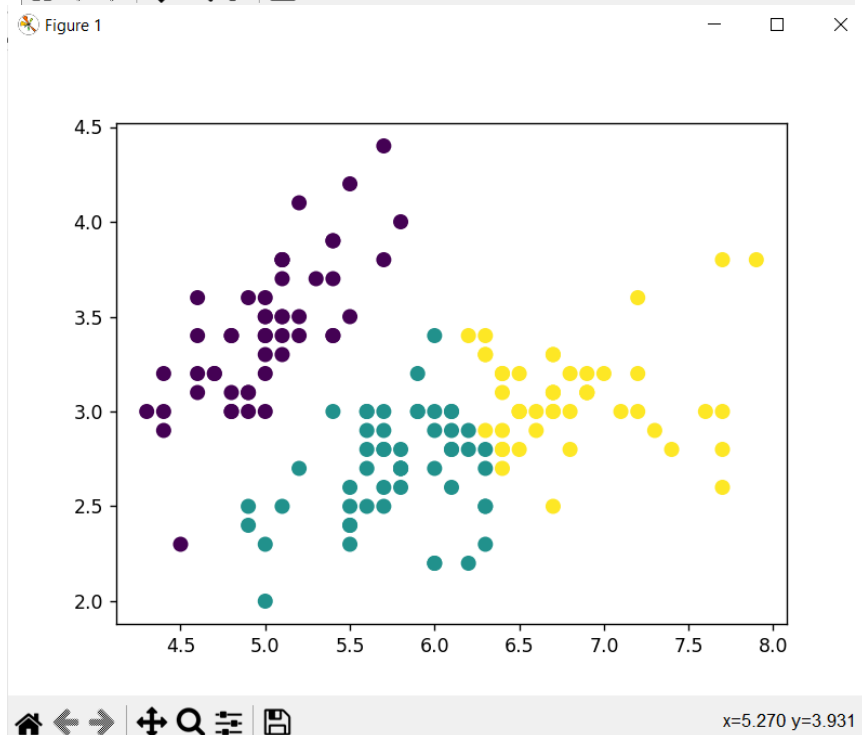
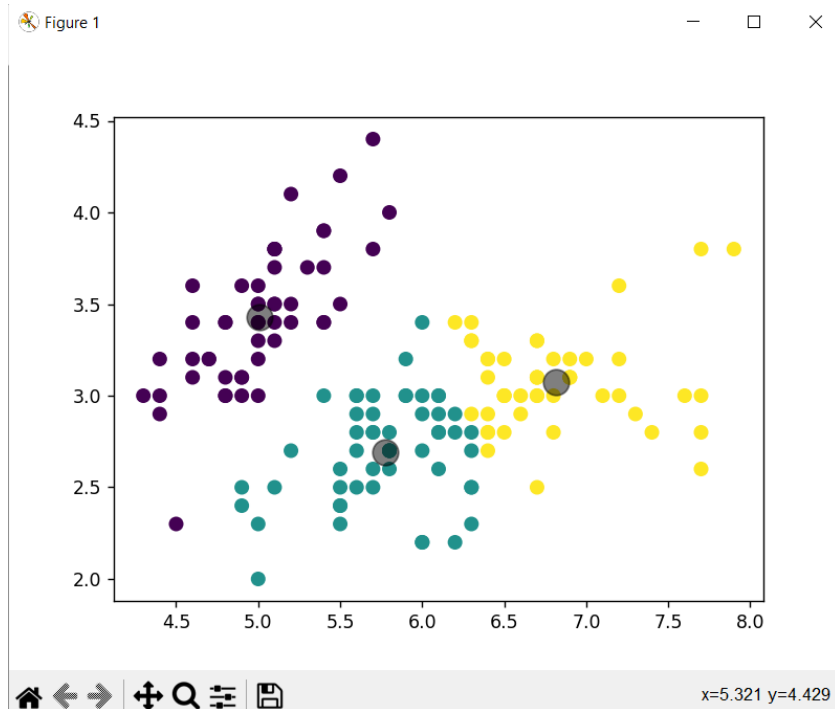
```

1 import matplotlib.pyplot as plt
2 from sklearn import datasets
3 from sklearn.cluster import KMeans
4 from sklearn.metrics import pairwise_distances_argmin
5 import numpy as np
6 # Отримуємо дані
7 iris = datasets.load_iris()
8 X = iris.data[:, :2]
9 Y = iris.target
10 # Визначаємо початкові кластери
11 kmeans = KMeans(n_clusters=Y.max() + 1, init='k-means++', n_init=10, max_iter=300,
12                 tol=0.0001, verbose=0, random_state=None, copy_x=True)
13 kmeans.fit(X)
14 y_pred = kmeans.predict(X)
15 print("n_clusters: 3, n_init: 10, max_iter: 300, tol: 0.0001, verbose: 0, random_state: None, copy_x: True")
16 print(y_pred)
17 plt.figure()
18 plt.scatter(X[:, 0], X[:, 1], c=y_pred, s=50, cmap='viridis')
19 centers = kmeans.cluster_centers_
20 plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)
21 plt.show()
22
23 def find_clusters(X, n_clusters, rseed=2):
24     # Випадково обираємо кластери
25     rng = np.random.RandomState(rseed)
26     i = rng.permutation(X.shape[0])[:n_clusters]
27     centers = X[i]
28
29     while True:
30         # Голошуємо label базуючись на найближчому центрі
31         labels = pairwise_distances_argmin(X, centers)
32
33         # Знаходимо нові центри з середини точок
34         new_centers = np.array([X[labels == i].mean(0) for i in range(n_clusters)])
35
36         # Перевірка збіжності
37         if np.all(centers == new_centers):
38             break
39         centers = new_centers
40     return centers, labels
41
42 print("using find_clusters():")
43 centers, labels = find_clusters(X, 3)
44 print("n_clusters: 3, rseed: 2")
45 plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
46 plt.show()
47
48 centers, labels = find_clusters(X, 3, rseed=0)
49 print("n_clusters: 3, rseed: 0")
50 plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
51 plt.show()
52
53 labels = KMeans(3, random_state=0).fit_predict(X)
54 print("n_clusters: 3, rseed: 0")
55 plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
56 plt.show()

```

Рис 15. Код програми файлу LR\_3\_task\_8

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				13
Змн.	Арк.	№ докум.	Підпис	Дата		



		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				14
Змн.	Арк.	№ докум.	Підпис	Дата		



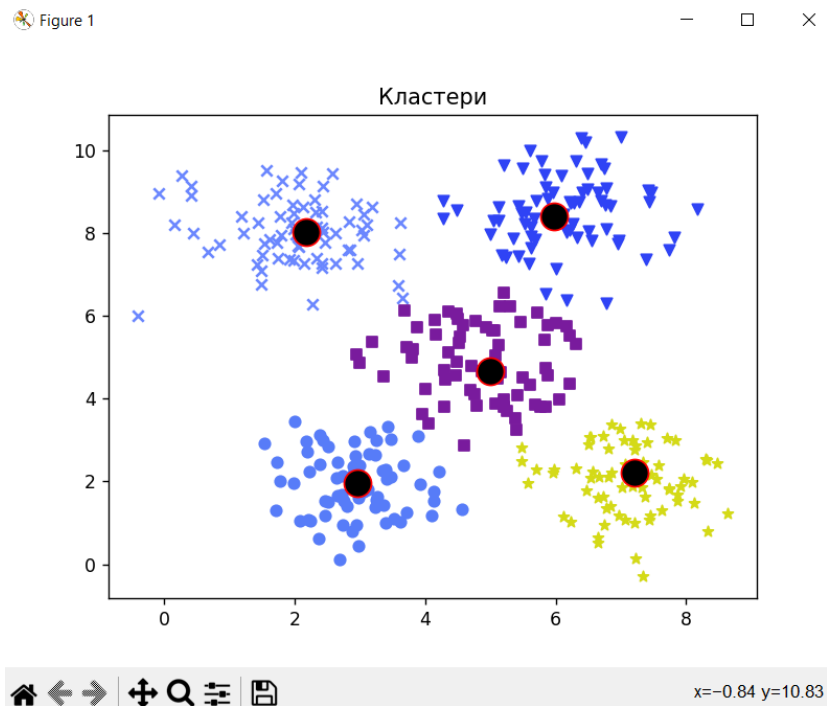


Рис 18. Результат виконання коду файлу LR\_3\_task\_9

Метод зсуву середнього – доволі валідний алгоритм, головною перевагою якого є непотрібність жодних припущень щодо базового розподілу даних, має змогу обробляти довільні простори функцій, проте важливу роль відіграє обрана ширина вікна (bandwidth).

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				16
Змн.	Арк.	№ докум.	Підпис	Дата		



**Висновок:** під час виконання лабораторної роботи я навчився: використовувати спеціалізовані бібліотеки і мову програмування Python, досліджувати методи регресії та неконтрольованої класифікації даних у машинному навчанні.

Посилання на гіт: <https://github.com/Max1648/Artificial-Intelligence2>

		Надворний М.Ю			ДУ «Житомирська політехніка».20.121.12 – Лр3	Арк.
		Філіпов В.О.				17
Змн.	Арк.	№ докум.	Підпис	Дата		